#13
1-15-02

PATENT APPLICATION

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re application of

William J. BAER et al.

Appln. No.: 09/220,293

Confirmation No.: 3693

Filed: December 2, 1998

Express Mail No. EF279045868US

Group Art Unit: 2171

Examiner: J. Veillard

For: METHOD FOR CONFIGURATION MAPPING BETWEEN DATA STORES AND DATA STRUCTURES AND A GENERALIZED CLIENT DATA MODEL USING HETEROGENEOUS, SPECIALIZED STORAGE REPRESENTATIONS

SUBMISSION OF APPELLANTS' BRIEF ON APPEAL

Commissioner for Patents
Washington, D.C. 20231

Sir:

Submitted herewith please find an original and two copies of Appellants' Brief on Appeal. Please charge the statutory fee of $320.00 to Deposit Account No. 19-4880. Authorization is also given to charge or credit any difference or overpayment to Deposit Account No. 19-4880. A duplicate copy of this paper is attached.

Respectfully submitted,

Frank L Bernstein

Frank L. Bernstein
Registration No. 31,484

SUGHRUE MION, PLLC
1010 El Camino Real, Suite 360
Menlo Park, CA 94025

Tel: (650) 325-5800
Fax: (650) 325-6606

Date: December 31, 2001

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of

William J. BAER et al.

Application No.: 09/220,293

Confirmation No.: 3693

Filed: December 23, 1998

Express Mail No. EF279045868US

Group Art Unit: 2175

Examiner: J. Veillard

FOR: METHOD AND APPARATUS FOR CONFIGURABLE MAPPING BETWEEN DATA STORES AND DATA STRUCTURES AND A GENERALIZED CLIENT DATA MODEL USING HETEROGENOUS, SPECIALIZED STORAGE

APPELLANTS' BRIEF ON APPEAL UNDER 37 C.F.R. §1.192

Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

Appellants, within a two (2) month period from the November 21, 2001, filing date of the Notice of Appeal, herein file an Appeal Brief drafted in accordance with the provisions of 37 C.F.R. §1.192, as follows:

I. REAL PARTY IN INTEREST

The real party in interest here is the owner of the application, IBM Corporation.

II. RELATED APPEALS AND INTERFERENCES

Appellants appealed on November 2, 2001 from the final Office Action dated August 2, 2001 in application No. 09/219,934 (hereinafter "the '934 application"). Appellants will appeal from the final Office Action dated December 12, 2001 in application No. 09/220,291 (hereinafter "the '291 application"). Both the '934 application and the '291 application have the same specification as the present application and were filed on the same day as the present application). The '934 application was rejected by one Examiner, who construes "transferring

data to a data store" in both Mullins and the '934 application as "querying." However, the present application and the '291 application were rejected by another Examiner, who construes "transferring data to a data store" in Mullins and the present application, but not the '291 application, as "writing data to a data store". To the best of their knowledge, Appellants are not aware of any interferences involving the present application.

III. STATUS OF CLAIMS

Claims 1-31 are pending in the present application. Claims 10-31 are allowed, while claims 1-5 stand rejected under 35 U.S.C. §102(e) as being anticipated by USP 5,857,197 (Mullins), and claims 6-9 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Mullins in view of USP 6,006,230 (Ludwig et al.)

IV. STATUS OF AMEDMENTS

The claims have not been amended pursuant to final rejection.

V. SUMMARY OF THE INVENTION

The present application provides a flexibly adaptable asset management system with read-write capability features for processing and manipulating assets. An asset of the present application is defined to be a set of related data, or meta data, for a document or an object and/or the actual data itself (page 5, lines 6-7). The set of related data may consist of relational data, files, references to files from indexing engines, or any other combination of data types (page 5, lines 9-10). The assets may represent, for example, data in the form of text, full-motion video, audio, graphics, or images (page 2, lines 14-15).

The system comprises three layers: a client application layer, for manipulating and browsing assets; an asset manager server layer; and a data store representing the third layer of the system. The asset manager server layer comprises several configurable modules, comprising a client adapter module, a schema adapter module, and a resources module. The asset manager server provides, for specified asset types, programming interface services, such as storing, querying, and retrieving assets representing data to and from the data store. Each schema adapter maps the data store schema into assets of a particular asset type, and new schema adapters are constructed to support database operations for new types of asset (page 13, lines 13-16, and page 14, lines 6-7).

The schema adapter (used interchangeably with "schema controller" in the specification) defines the communication between the asset manager server and the data store, and is also responsible for implementing the interfaces defined by the client adapter, e.g., query, result, edit and system administrator (page 14, lines 4-6). Information shared between the client application and schema adapter can be either returning data to a client application or to update or add data to the data store (page 19, lines 29-31). In this way, the schema adapter of the present application transfers data to and from the data store in response to methods invoked in the client adapter by the client application, as recited in claims 1-5 and 6-9. Users can add data to the data store, and update, delete, retrieve and query data in the data store. The system of the present application can also be used to transfer information between data stores. Therefore, "transferring data to a data store" in the present application means "writing data to a data store."

The prior art provides a system for querying databases located on multiple platforms, processing data obtained, and presenting processed data to a user. The prior art, however, fails to disclose a flexibly adaptable asset management system with read-write capability features of the present application.

VI. ISSUES

1. Does the prior art teach or reasonably suggest the flexibly adaptable asset management system with read-write capability features, as recited in claim 1?

2. Does the prior art teach or reasonably suggest the provision of a schema adapter that is specific to a particular one of the assets, as recited in claim 2?

3. Does the prior art teach or reasonably suggest an instance of an object oriented class that encapsulates the data and associated behaviors for transferring between the at least one schema adapter and the client application through the at least one client adapter, as recited in claim 3?

4. Does the prior art teach or reasonably suggest the provision of external services for providing a link between the at least one schema adapter and the data store for transferring data to and from the data store, as recited in claim 4?

5. Does the prior art teach or reasonably suggest registration of a schema adapter with the asset manager server by identifying a client adapter supported by the schema adapter, and implementation of the interface functions defined in the supported client adapter by the schema adapter, as recited in claim 5?

6. Does the prior art teach or reasonably suggest the identification of a schema adapter by a unique identifier, as recited in claim 6?

7. Does the prior art teach or reasonably suggest the identification of an asset type by a unique identifier, as recited in claims 7 and 8?

8. Does the prior art teach or reasonably suggest implementation a parser for extracting properties and associated values from files stored in the data store, as recited in claim 9?

VII. GROUPING OF CLAIMS

Each of claims 1-6 and 9 stands and falls separately. Claims 7 and 8 stand and fall together.

VIII. ARGUMENT

Issue 1

Introduction

The present invention significantly differs from Mullins. The Mullins system and method cannot be used for the claimed purpose of the present application, because Mullins only teaches reading data from a data store, but and not writing data to the data store.

The Examiner in charge of the present application is also in charge of the '291 application. After a telephone interview with Appellants on September 11, 2001, the Examiner appeared to agree that the present application discloses and recites an asset management system with read-write capability features, and that "transferring data to a data store" in the present application and in the '291 application means "writing data to a data store".

However, the Examiner also construes "transferring data to a data store" in Mullins as "writing data to a data store," the same meaning as "transferring data to a data store" in the present application and in the '291 application, and consequently maintains that Mullins teaches

the read-write capability features of the present application. The Examiner's interpretation of "transferring data to a data store" in Mullins is different from that of the Examiner in the '934 application, who construes the term in Mullins as meaning "querying".

Mullins

Mullins discloses a method and system for querying data stored in data stores, which could be either an object or a non-object data store, as objects from an object application. Specifically, Mullins seeks to solve a data store access problem which arises when two or more data stores, which have different underlying structures, are accessed for stored data. For example, in the system shown in Fig. 1 of Mullins, an object application 101 accesses both object data store 312 and non-object data store 302 for stored data.

The Mullins system comprises an adapter abstraction layer 600 which further comprises a first adapter 400 and a second adapter 500. The adapter abstraction layer 600 performs the necessary conversions between object and non-object views, and consequently allows both object data store and non-object data store to be accessed identically for stored data from at least one object application.

In describing this system, Mullins states that a request 100 and an accompanying object 102 are passed from an application program to the first adapter 400. The first adapter 400 then extracts object attributes 103 and the object name 104 from the object 102, packs object attributes 103 and the object name 104 as "data" 105, and communicates the "data" 105 and the request 100 to the second adapter 500 (Mullins at col. 7, lines 39-54). The second adapter 500 searches a meta data map using the object name 104, and generates a command 303, using the

object attributes 103, for accessing the data store 302 according to the request 100 (Mullins at col. 7, lines 55-67). The second adapter 500 then executes command 303, obtains and processes the data store content 304 from data store 302, packs the obtained data store content 304 and the execution status as data 115, and communicates the data 115 to the first adapter 400 (Mullins at col. 8, lines 1-26).

Mullins names what is communicated from the first adapter to the second adapter as "data", but looking at this so-called "data" more closely, one can see that this "data is basically a command or a query, dramatically distinguished from "data" as described in the present application, which consists relational data, files, references to files from indexing engines, or any other combination of data types (page 5, lines 9-10).

It is clear that nothing is written to Mullins' data stores through the communication from the first adapter to the second adapter, and then to the data stores. In addition, Mullins specifically states that the use of its technology provides "read only" data stores over the Internet (Mullins at col. 7, lines 64-66).

Although "transferring the requested data store content from the first adapter" is mentioned in Mullins (Mullins at col. 4, lines 49-65), this transfer is not described in any detail in Mullins' specification, and moreover is contradicted by other discussion in Mullins, for example, at the bottom of column 7 of Mullins, which talks about a "read-only" data store. Thus, Appellants submit that the Mullins description on which the Examiner relies is not enabling.

Therefore, Mullins teaches a read-only system for reading data stored in data stores which have different underlying structures, but lacks disclosure of read-write capability features of the present application. Communication of data from the first adapter to the second adapter is not a transfer, as disclosed and claimed in the present application. "Transferring data to a data store" in the present application means "writing data to a data store". The ordinarily skilled artisan would interpret "transferring" data "to" a place, such as a data store, as writing data to the data store. The ordinarily skilled artisan would not interpret the transfer of data to a data store as broadly as to encompass querying. The Mullins system and method thus are no more than what Appellants acknowledge as prior art.

From the foregoing, it is clear that Mullins discloses at most "querying" by communicating a command. Moreover, Mullins discloses a read-only system. Pursuant to the foregoing discussion, Appellants submit that claim 1 in the subject application is patentable.

Issue 2

With regard to claim 2, Appellants submit that, for all the reasons which have been discussed in detail above, claim 2 is patentable at least by virtue of its dependence on the patentable independent claim 1.

Furthermore, Mullins discloses a method and system for querying data stored in data stores having different underlying structures, e.g., object data store 312 and non-object data store 302. The adapter abstraction layer 600 performs the necessary conversions between object and non-object views, and therefore allows both object data store and non-object data store to be accessed identically from at least one object application (Mullins, col. 4, lines 9-14). The

abstraction layer 600 is applicable to data stores having different data structures, and the adapters can be transparently interchanged by the object application (Mullins, col. 9, lines 34-38). Mullins does not teach a schema adapter that is specific to a particular one of the assets, as claimed in the present application. Accordingly, Appellants submit that claim 2 is patentable for this additional reason as well.

Issue 3

With regard to claim 3, Appellants submit that, for all the reasons which have been discussed in detail above, claim 3 is patentable at least by virtue of its dependence on the patentable independent claim 1.

Furthermore, Mullins discloses a method and system for querying data stored in data stores having different underlying structures. As each object data is being read from the data stores, an instance of the object's class type is instantiated and initialized (Mullins, col. 8, lines 21-26). The instance of the object oriented class in Mullins is used to transfer obtained data from the data stores only, while the instance of the object oriented class claimed in the present application also transfers assets to be written to the data stores. Therefore, claim 3 is patentable for this additional reason as well.

Issue 4

With regard to claim 4, Appellants submit that, for all the reasons which have been discussed in detail above, claim 4 is patentable at least by virtue of its dependence on the patentable independent claim 1.

Furthermore, Mullins teaches a read-only system for querying data stored in data stores having different underlying structures. Nothing is written to Mullins' data stores through the communication from the first adapter to the second adapter, and then to the data stores. What is communicated along the link between the data stores and the second adapter 500 of the Mullins system includes commands from the second adapter 500 and stored data from the data stores, but does not include assets to be written to the data stores as claimed in the present application. Appellants submit that claim 4 is patentable for this additional reason as well.

Issue 5

With regard to claim 5, Appellants submit that, for all the reasons which have been discussed in detail above, claim 5 is patentable at least by virtue of its dependence on the patentable independent claim 1.

Furthermore, in the Mullins system, the adapter abstraction layer 600 includes a first adapter 400 and a second adapter 500. The first adapter 400 extracts object attributes 103 and the object name 104 from the object 102, packs object attributes 103 and the object name 104 as "data" 105, and communicates the "data" 105 and the request 100 to the second adapter 500 (Mullins at col. 7, lines 39-54). The second adapter 500 generates a command 303 and executes it to access the data stores according to request 100 (Mullins at col. 7, lines 55-67). Mullins does not teach or even suggest that the second adapter 500 registers with the adapter abstraction layer 600 by identifying the first adapter 400, and that the first adapter 400 defines an interface to be implemented by the second adapter 500. Appellants submit that claim 5 is patentable for these additional reasons as well.

Issue 6

As discussed above, Mullins does not teach a flexibly adaptable asset management system with read-write capability features disclosed and claimed in the present application. Accordingly, claim 6 is patentable at least by virtue of its dependence on the patentable independent claim 1.

The Examiner has asserted that claim 6 is unpatentable over Mullins in view of Ludwig et al., which teaches a database client/server development system with the limitations of "wherein each of the at least one client adapter is identified by a unique identifier". Even assuming *arguendo* the Mullins could be construed as teaching the read-write capability features of the present application, Ludwig et al. does not teach identification of a client adapter by a unique identifier.

Ludwig et al.

Ludwig et al. discloses a database client server development system providing support for remote sessions with user-created application objects, and in particular a system allowing a programmer to create objects that contain business rules and which can be distributed onto one or more application servers. The client/server database system shown in Fig. 2 of Ludwig et al. comprises one or more client(s) 210 connected to a server 230 via a network 220. The server 230 maintains one or more database indexes on tables 250. An index may be constructed as a single disk file storing index key values together with unique record numbers. The latter are unique pointers or identifiers to the actual storage location of each record in the database file (Ludwig et al., col. 7, lines 48-61). The server 230 includes engine 260, including a parser 261.

Although Ludwig et al. discusses "unique pointers and identifiers", these identifiers relate to the database storage location of a record (Ludwig at col. 7, lines 59-61). It is clear that the utilization of "unique pointers and identifiers" in Ludwig et al. and that in the present application significantly differ from each other: the former is to identify database storage location of a record, while the latter is to identify a client adapter or an asset type.

Therefore, Ludwig et al. does not teach identification of a client adapter by a unique identifier. Appellants submit that claim 6 in the subject application is patentable for this additional reason.

Issue 7

With regard to claims 7 and 8, Appellants submit that, for all the reasons which have been discussed in detail above, claims 7 and 8 are patentable at least by virtue of their dependence on the patentable independent claim 1.

Furthermore, Ludwig et al. utilizes "unique pointers and identifiers" to identify the database storage location of a record, instead of an asset type. Accordingly, Appellants submit that claims 7 and 8 are patentable for this additional reason as well.

Issue 8

With regard to claim 9, Appellants submit that, for all the reasons which have been discussed in detail above, claims 9 is patentable at least by virtue of their dependence on the patentable independent claim 1.

Furthermore, the parser 261 in Ludwig et al. is used to convert SQL (Structured Query Language) statements into a query tree-a binary tree data structure which represents the

components of the query in a format selected for the convenience of the system (Ludwig et al., col. 8, lines 6-10), instead of extracting property information from a file as claimed in the present application.

In addition, Mullins teaches a read-only system for querying data stored in data stores having different underlying structures, but lacks disclosure of read-write capability features of the present application. Nothing is written to Mullins' data stores through the communication from the first adapter to the second adapter, and then to the data stores. Although Ludwig et al. mentions that the engine 260 of the database server system 240 comprises a parser 261 (Ludwig et al., col. 8, lines 3-6), the combination of Mullins and Ludwig et al. would not result in the parser of the present application, which is used to extract properties and their values from a file or stream to populate the data store 40 (page 22, lines 4-5 and 10-11).

Therefore, Appellants submit that claim 9 is patentable for these additional reasons as well.

IX. CONCLUSION

Pursuant to the foregoing arguments, Appellants submit that claims 1-31 in the subject application are patentable.

The present Brief on Appeal is being filed in triplicate. Appellants hereby petition for any extension of time which may be required to maintain the pendency of this case, and any required fee for such extension is to be charged to Deposit Account No. 19-4880.

Respectfully submitted,

Frank L. Bernstein
Registration No. 31,484

SUGHRUE MION, PLLC
1010 El Camino Real, Suite 360
Menlo Park, CA 94025

Tel: 650-325-5800
Fax: 650-325-6606

Date: December 31, 2001

APPENDIX

Claims on Appeal: 1-9.

1. A flexibly adaptable asset management system for deploying asset management functions to a client application for manipulating assets, representing data, in a data store, and for adaptively mapping between the assets and the data store, the system comprising:

an asset manager server disposed between the client application and the data store, the asset manager server including:

at least one client adapter for providing interface functions between the client application and the asset manager server; and

at least one schema adapter for mapping the assets to the data stored in the data store and for transferring the data to and from the data store in response to methods invoked in the at least one client adapter of the client application,

wherein, the at least one schema adapter is flexibly adaptable, thereby allowing the system to do one or more of handle different asset types and handle additional client applications.

2. The system according to claim 1, wherein the at least one schema adapter is specific to a particular one of the assets, an asset being meta data for a particular data type.

3. The system according to claim 1, wherein the asset manager server further includes:

at least one object oriented class, wherein an instance of the object oriented class encapsulates the data and associated behaviors for transferring between the at least one schema adapter and the client application through the at least one client adapter.

4. The system according to claim 1, further comprising external services for providing a link between the at least one schema adapter and the data store.

5. The system according to claim 1, wherein the at least one schema adapter registers with the asset manager server by identifying ones of the at least one client adapter supported by the at least one schema adapter, wherein the at least one schema adapter implements the interface functions defined in the supported client adapter.

6. The system according to claim 1, wherein the at least one schema adapter is identified by a unique identifier.

7. The system according to claim 1, wherein the at least one schema adapter supports an asset type, identified by a unique identifier, which is associated with the particular one of the assets and corresponds to a file type.

8. The system according to claim 7, wherein the at least one schema adapter supports multiple asset types, each of the asset types being identified by a unique identifier.

9. The system according to claim 1, further comprising implementing a parser for extracting properties and associated values from files stored in the data store.

01-07-01

AF

2171

PATENT APPLICATION

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of

William J. BAER et al.

Application No.: 09/220,293

Group Art Unit: 2175

Confirmation No.: 3693

Examiner: J. Veillard

Filed: December 23, 1998

FOR: METHOD AND APPARATUS FOR CONFIGURABLE MAPPING BETWEEN DATA STORES AND DATA STRUCTURES AND A GENERALIZED CLIENT DATA MODEL USING HETEROGENOUS, SPECIALIZED STORAGE

"EXPRESS MAIL" CERTIFICATE OF MAILING

I hereby certify that this paper and the attachments hereto are being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated below, addressed to the Commissioner for Patents, Washington, DC 20231.

Express Mail Certificate Number: EF279045868US

EF 279045868 US

Date: December 31, 2001

Signed: Thea K. Wagner

Thea K. Wagner

Enclosures:

1. Submission of Appellants' Brief on Appeal (in duplicate);
2. Appellants' Brief on Appeal (in triplicate);
3. Postcard filing receipt.